Opacity Enforcing Control Synthesis

Jérémy Dubreil, Philippe Darondeau, Hervé Marchand

I R I S A
INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTÈMES ALÉATOIRES

PUBLICATION
INTERNE
N° 1887

OPACITY ENFORCING CONTROL SYNTHESIS

JÉRÉMY DUBREIL, PHILIPPE DARONDEAU AND HERVÉ MARCHAND

ISSN 1166-8687

IRISA
CAMPUS UNIVERSITAIRE DE BEAULIEU - 35042 RENNES CEDEX - FRANCE

IRISA

Institut de Recherche en Informatique et Systèmes Aléatoires
Campus de Beaulieu – 35042 Rennes Cedex – France
Tél. : (33) 02 99 84 71 00 – Fax : (33) 02 99 84 71 71
http://www.irisa.fr

Opacity Enforcing Control Synthesis

Jérémy Dubreil, Philippe Darondeau and Hervé Marchand

Systèmes communicants
Projets VerTeCs-S4

Publication interne n ° 1887 — March 2008 — 17 pages

Abstract: Given a finite transition system and a regular predicate, we address the problem of computing a controller enforcing the opacity of the predicate against an attacker (who partially observes the system), supposedly trying to push the system to reveal the predicate. Assuming that the controller can only control a subset of the events it observes (possibly different from the ones of the attacker), we show that an optimal control always exists and provide sufficient conditions under which it is regular and effectively computable. These conditions rely on the inclusion relationships between the controllable alphabet and the observable alphabets of the attacker and of the controller.

Key-words: control, security, opacity, discrete event systems, partial observation.

(Résumé : tsvp)

Centre National de la Recherche Scientifique
(UMR 6074) Université de Rennes 1 – Insa de Rennes

Institut National de Recherche en Informatique
et en Automatique – unité de recherche de Rennes

Assurer l'opacité par synthèse de contrôleur

Résumé : Étant donné un système de transitions fini et un prédicat régulier, nous nous intéressons à la construction d'un contrôleur assurant l'opacité de ce prédicat vis à vis d'un attaquant (observant partiellement le système), qui force le système à révéler le prédicat. En supposant que le contrôleur peut seulement contrôler un sous-ensemble des événements qu'il observe (potentiellement différents de ceux observés par l'attaquant), nous montrons qu'un contrôle optimal existe et donnons des conditions suffisantes sous lesquelles la solution est régulière et effectivement calculable. Ces conditions concernent les relations d'inclusion entre les alphabets du contrôleur et de l'attaquant et l'alphabet des événements contrôlables.

Mots clés : contrôle, sécurité, opacité, systèmes à événements discrets, observation partielle.

1 Introduction

Opacity, whose goal is to oppose diagnosis, was introduced in [8] and [4]. Given a system, equipped with a map sending (prefixes of) executions to observations, an opaque predicate is a set of executions such that every execution in the set is observationally equivalent to some execution outside the set. So, membership to an opaque predicate is never disclosed by observation. Anonymity and non-interference may be reduced to the opacity of suitable predicates for suitable observation maps [4]. In this paper, we concentrate on finite transition systems labelled over an alphabet Σ, on predicates defined by regular sets of execution traces in Σ^*, and on observation maps induced by the projection of execution traces on a sub-alphabet Σ_a of Σ, modeling the attacker's alphabet. Under these assumptions, opacity can be decided although it cannot be expressed in the modal μ-calculus [1].

We are specially interested in cases when the predicate of interest is *non-opaque*, i.e. the system leaks confidential information. A possible arrangement is then to augment the system with a monitor, responsible for detecting when confidential information was leaked or will be leaked unless one halts the system immediately. Assuming that monitors observe only a subset Σ_m of the events of the system, which needs not be a subset of Σ_a, necessary and sufficient conditions for the existence of monitors were obtained in [6]. We want to take one step further by providing a controller that does enforce the opacity of the predicate by disabling at each stage (of an execution) the least subset of events such that confidential information is not leaked sooner or later. Assuming that controllers observe all events and all events can be controlled, sufficient conditions for the existence of finite state controllers were proposed in [2] (the opacity of several predicates is enforced there on concurrent attackers). We consider here one predicate and one attacker, but we relax the assumptions on controllable and observable events. Namely, if Σ is the set of events of the system, let $\Sigma_a \subseteq \Sigma$ be the attacker's alphabet, and let Σ_c and Σ_m be the subsets of events controlled or observed by the controller, respectively, then we assume that $\Sigma_c \subseteq \Sigma_m$ and Σ_a compares both with Σ_c and Σ_m.

Let $L(G) \subseteq \Sigma^*$ be the regular language of the system G and let $L_\varphi \subseteq \Sigma^*$ be the regular but non-opaque predicate whose opacity should be enforced by control. Not taking into account controllability and observability, there is a largest subset L_1 of $L(G)$ such that L_φ is opaque w.r.t. L_1 and Σ_a, and L_1 is regular [2]. As $\Sigma_c \subseteq \Sigma_m$, there exists a most permissive controller K_1 confining the system to L_1 and K_1 is regular. Unfortunately, this controller *does not always enforce the opacity of* L_φ (unless $\Sigma_a \subseteq \Sigma_c$ or $\Sigma_m \subseteq \Sigma_a$ as we shall explain later on). The reason why it fails to do so is that a complete description of the closed-loop system may be available to the attacker and new confidential information on the execution may be inferred from this knowledge. To solve the problem, one might think of iterating the construction, thus producing a decreasing chain of regular languages $L(G) = K_0 \supseteq K_1 \supseteq K_2 \supseteq \ldots$. Unfortunately, the iteration may be infinite, hence it may not yield an effective construction of $\cap_i K_i$ and it does not show either that this limit is regular.

Our contribution is twofold. For the cases $\Sigma_a \subseteq \Sigma_c$ and $\Sigma_m \subseteq \Sigma_a$, we show that the optimal opacity control can be computed within the framework of Ramadge and Wonham's theory. For the remaining case $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_m$[1], for which the iteration may be infinite, we supply an alternative algorithm that computes the limit of the infinite iteration described above. The algorithm works in double exponential time. We do not investigate optimizations nor heuristics in this paper for our primary goal is to show that the construction of the optimal opacity control is effective.

This work has loose relationship with the earlier work done by Schneider on security automata [9], subsequently extended to edit automata [7]. The goal pursued in [9] was to produce interface automata that enforce *security policies* L_φ , meaning that the interface automaton rejects those inputs from the environment that would lead the system to leave the subset of safe execution prefixes L_φ. In our case, the role of the controller is not to confine the executions of the system to L_φ but to the largest opaque subset of $L(G)$ w.r.t. L_φ and Σ_a. On the other hand, whenever $\Sigma_c \subseteq \Sigma_a$, our controllers may be seen as interface automata, as they reject events from the attacker's alphabet exclusively.

[1] Recalling that Σ_a compares with both Σ_c and Σ_m.

The rest of the paper is organized as follows. Section 2 fixes some notation. Section 3 brings back the basics of opacity properties and it sets the opacity control problem. Section 4 brings back the theory of Supervisory Control. Section 5, which is the core of the paper, contains our contribution. Optimal opacity control is obtained whenever $\Sigma_c \subseteq \Sigma_m$ and Σ_a compares with both of them. Moreover, we produce an example showing that the problem cannot be solved in the framework of Ramadge and Wonham's theory when $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_m$. Section 6 is a brief conclusion pointing to open problems.

2 Notations

Let Σ be a finite alphabet of events. A *string* is a finite sequence of events. The set of all strings is denoted by Σ^*. Any subset of Σ^* is called a *language* over Σ. Let L be a language over Σ. The *prefix-closure* of L is defined as $\overline{L} = \{s \in \Sigma^* \mid \exists t \in \Sigma^* \text{ s.t. } st \in L\}$. We assume that systems are Labelled Transitions Systems (LTS) as follows.

Definition 1 (LTS) *An LTS over Σ is a 4-tuple $G = (Q_G, \Sigma, \delta_G, q_0^G)$ where Q_G is a finite set of states, Σ is the finite set of* events *of G, $q_0^G \in Q_G$ is the initial state, and $\delta_G : Q_G \times \Sigma \to Q_G$ is* a partial transition function. •

In the sequel, we write $q \xrightarrow{a}_G q'$ if $\delta(q,a) = q'$ and $q \xrightarrow{a}_G$ if $\exists q' \in Q_G,\ q \xrightarrow{a}_G q'$. We extend $\to_{\mathrm{G}}$ to arbitrary sequences by setting $q \xrightarrow{\varepsilon}_{\mathrm{G}} q$ for all states q, and $q \xrightarrow{s\sigma}_{\mathrm{G}} q'$ whenever $q \xrightarrow{s}_{\mathrm{G}} q''$ and $q'' \xrightarrow{\sigma}_{\mathrm{G}} q'$, for some $q'' \in Q_{\mathrm{G}}$, $s \in \Sigma^*$ and $\sigma \in \Sigma$. We denote

$$\mathcal{T}_G = \{(q, \sigma, q') \in Q \times \Sigma \times Q \ :\ q \xrightarrow{\sigma} q'\}$$

the set of transitions of G and $L(G) = \{l \in \Sigma^* \mid q_0^{\mathrm{G}} \xrightarrow{l}_G\}$ the set of its execution traces. Given non-empty subsets $I_{\mathrm{G}}, F_{\mathrm{G}} \subseteq Q_G$, the definitions extend to $L_{F_G}(G) = \{s \in \Sigma^* \mid \exists q \in F_G, q_0^{\mathrm{G}} \xrightarrow{s}_G q\}$ (the set of execution traces ending in a final state of F_G) and $L_{I_G,F_G}(G) = \{s \in \Sigma^* \mid \exists q' \in I_G,\ \exists q \in F_G, q' \xrightarrow{s}_G q\}$ (the set of partial execution traces starting in a state of I_G and ending in a state of F_G).

Opacity control aims at preventing an attacker $\mathcal{A}$ from deducing confidential information on the execution of a system from the observation of a subset of events Σ_a. To model this, we use the classical notion of *projection*. Let $P_{\Sigma_a} : \Sigma^* \to \Sigma_a^*$ be the natural *projection* of execution traces onto Σ_a^* defined by: $P_{\Sigma_a}(\epsilon) = \epsilon$ and $P_{\Sigma_a}(s\sigma) = P_{\Sigma_a}(s).\sigma$ if $\sigma \in \Sigma_a$, and $P_{\Sigma_a}(s)$ otherwise. The projection simply erases in a sequence of Σ^* all events not in Σ_a. This definition extends to (regular) languages:

$$P_{\Sigma_a}(K) = \{\mu \in \Sigma_a^* \mid \exists s \in K,\ \mu = P_{\Sigma_a}(s)\}.$$

Conversely, given $K \subseteq \Sigma_a^*$, the inverse projection of K is

$$P_{\Sigma_a}^{-1}(K) = \{s \in \Sigma^* \mid P_{\Sigma_a}(s) \in K\}$$

Given an LTS G over Σ and a set of observable events $\Sigma_a \subseteq \Sigma$, the set of *observed traces* of G is $P_{\Sigma_a}(L(G))$. Given two sequences $s, s' \in \Sigma^*$, we let $s \sim_a s'$ in case $P_{\Sigma_a}(s) = P_{\Sigma_a}(s')$ and denote $[s]_a = P_{\Sigma_a}^{-1}(P_{\Sigma_a}(s))$ the equivalence class of s.

Lemma 1 *Let $\Sigma_a \subseteq \Sigma_b \subseteq \Sigma$, then $s \sim_b s' \Rightarrow s \sim_a s'$.* ◇

3 The basics of opacity

Consider an LTS G over Σ, a regular predicate $L_\varphi \subseteq \Sigma^*$, and a sub-alphabet $\Sigma_a \subseteq \Sigma$. The alphabet Σ_a defines the interface provided to the user for interacting with G. The predicate L_φ represents a confidential information on the execution of G, i.e. if the current trace of execution is $s \in \Sigma^*$, the user should not be able to deduce from $P_{\Sigma_a}(s)$ and G that $s \in L_\varphi$. In this setting, the user is considered as

an attacker ($\mathcal{A}$) willing to catch the confidential information and armed for this with full information on the structure of G but only partial information upon its behavior, namely the observed trace in Σ_a^*. In order that the confidential information is never leaked, it is necessary and sufficient that L_φ is an opaque predicate according to the following definition, adapted from [4].

Definition 2 (Opacity) *L_φ is said to be opaque w.r.t. $L(G)$ and Σ_a if*

$$\forall s \in L(G),\ [s]_a \cap L(G) \not\subseteq L_\varphi \tag{1}$$

•

In other words, L_φ is opaque w.r.t. $L(G)$ and Σ_a if and only if

$$\forall \mu \in P_{\Sigma_a}(L(G)),\ P_{\Sigma_a}^{-1}(\mu) \cap L(G) \not\subseteq L_\varphi,$$

and L_φ is non-opaque w.r.t. $L(G)$ and Σ_a if and only if

$$\exists \mu \in P_{\Sigma_a}(L(G)),\ P_{\Sigma_a}^{-1}(\mu) \cap L(G) \subseteq L_\varphi.$$

Example 1 *Consider the two specifications G_1 and G_2 of a coffee-machine depicted in Figures 1 and 1, and let $\Sigma_a = \{coinIn, coinOut, cancel, confirm, coffeeOut\}$. Consider the predicate $L_\varphi = \Sigma^*.\underline{full}.\Sigma^*$. Then, L_φ is not opaque with respect to $L(G_1)$ and Σ_a, since e.g. for the observed trace*

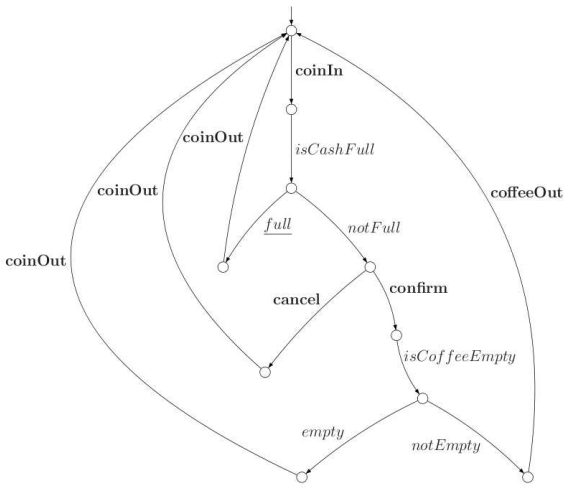


Figure 1: The predicate L_φ is non opaque w.r.t. G_1 and Σ_a

$coinIn.coinOut$ for which the only possible execution trace is $coinIn.isCashFull.\underline{full}.coinOut \in L_\varphi$. A contrario L_φ is opaque with respect to $L(G_2)$ and Σ_a. ◇

If L_φ is not opaque w.r.t. $L(G)$ and Σ_a, then it is still possible to restrict the behavior of G so that L_φ becomes opaque. This can be obtained by withdrawing from $L(G)$ all words $u.v$ such that $u \sim_a u' \Rightarrow u' \in L_\varphi$ for all $u' \in L(G)$.

Proposition 1 ([2]) *Given a system G and a predicate L_φ, there exists a supremal prefix-closed sublanguage of $L(G)$, noted $\text{OP}^\uparrow(L(G), L_\varphi, \Sigma_a)$, such that L_φ is opaque w.r.t. $\text{OP}^\uparrow(L(G), L_\varphi, \Sigma_a)$ and Σ_a, and it is given by*

$$\text{OP}^\uparrow(L(G), L_\varphi, \Sigma_a) = L(G) \setminus ((L(G) \setminus P_{\Sigma_a}^{-1}(P_{\Sigma_a}(L(G) \setminus L_\varphi))).\Sigma^*) \tag{2}$$

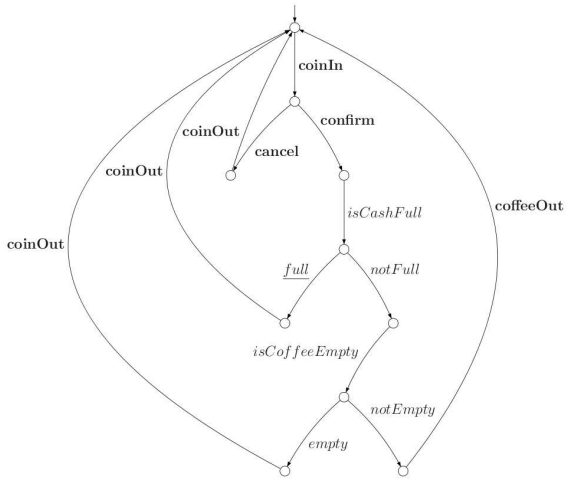


Figure 2: The predicate L_φ is opaque w.r.t. G_2 and Σ_a

Intuitively, the language $P^{-1}_{\Sigma_a}(P_{\Sigma_a}(L(G) \setminus L_\varphi))$ is the set of the "safe" sequences that do not reveal L_φ[2], whereas any sequence in $L(G) \setminus P^{-1}_{\Sigma_a}(P_{\Sigma_a}(L(G) \setminus L_\varphi))$ reveals L_φ (these sequences are extended with Σ^* because, once L_φ has been revealed, this holds for ever).

It follows from proposition 1 that $\text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$ is the union of all sub-languages L' of $L(G)$, such that L_φ is opaque w.r.t. L' and Σ_a [2]. Therefore, $\text{Op}^\uparrow$ is monotone in the first argument. Note that $\text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$ can be empty. In that case, there is no way to enforce opacity by restricting the behavior of the system.

Remark 1 *If L_φ is opaque w.r.t. L_1 and L_2, then it is opaque w.r.t. $L_1 \cup L_2$, but not necessarily w.r.t. $L_1 \cap L_2$. Similarly, if $L_1 \subseteq L \subseteq L_2$, L_φ may be opaque w.r.t. L but not opaque w.r.t. L_1 or L_2.*

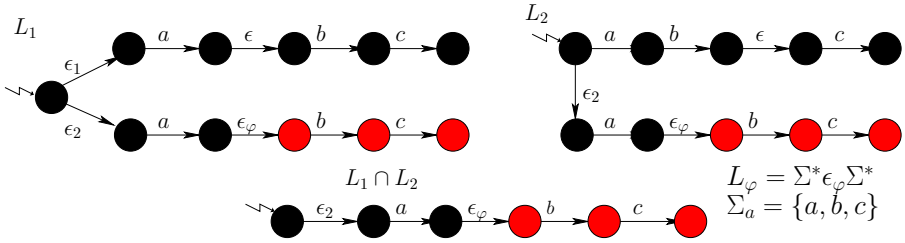


Figure 3: Intersection does not preserve opacity

•

Next, we establish a helpful lemma, stating that if a sequence s belongs to $\text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$, then any sequence in $L(G)$ observationally equivalent to s also belongs to $\text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$.

Lemma 2 $\forall s \in \text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a),\ [s]_a \cap L(G) \subseteq \text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$

Proof Let $s' \in [s]_a \cap L(G)$, then by definition $s \sim_a s'$ and $s' \in L(G)$. Suppose for a contradiction that $s' \notin \text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$, then $s' = uv$ for some u such that $u \sim_a u' \Rightarrow u' \in L_\varphi$ for all $u' \in L(G)$. As $s \sim_a s'$, $s = u"v"$ for some $u" \in L(G)$ such that $u \sim_a u"$. Therefore, $u" \sim_a u' \Rightarrow u' \in L_\varphi$ for all $u' \in L(G)$, showing that $s \notin \text{Op}^\uparrow(L(G), L_\varphi, \Sigma_a)$, a contradiction. ◇

[2]Note that this language is not prefix-closed.

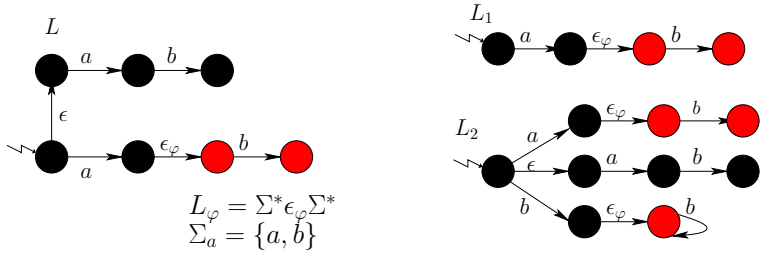


Figure 4: Inclusion and extension do not preserve opacity

Dually, this lemma implies that if a sequence s belongs to $L(G) \setminus \text{OP}^{\uparrow}(L_\varphi, L(G), \Sigma_a)$, then no sequence observationally equivalent to s belongs to $\text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a)$. In other words, when computing the supremal sub-language of $L(G)$ w.r.t. which L_φ is opaque, each equivalence class of $L(G)$ w.r.t. Σ_a is either entirely kept or removed.

Our goal is to enforce opacity by supervisory control, which puts strong conditions on the admissible restrictions of $L(G)$ (due to the so-called controllability and observability conditions that a controller has to fulfill to be implementable). We will also compute the most permissive opacity control in the form of a regular sub-language of $L(G)$. Next section brings back a few notions of supervisory control theory.

4 The Basics of Supervisory Control

Given a prefix-closed behavior $K \subseteq L(G) \subseteq \Sigma^*$ expected from the system G, the goal of supervisory control is to enforce this behavior on G by pairing this system with a monitor (also called controller) that observes a subset Σ_m of the events in Σ and controls a subset Σ_c of the events in Σ, i.e. enables or disables each instance of these controllable events. $\Sigma \setminus \Sigma_c$ is the set of uncontrollable events. $\Sigma \setminus \Sigma_m$ is the set of unobservable events. We now recall some basic concepts of supervisory control theory. More information on the computational aspects can be found in [5].

Definition 3 *A prefix-closed language $K \subseteq L(G)$ is controllable w.r.t. $L(G)$ and Σ_c if $K.(\Sigma \setminus \Sigma_c) \cap L(G) \subseteq K$.* •

This definition states that if K is controllable, then no uncontrollable events need to be disabled to exactly confine the system $L(G)$ to K. Note that the union of an arbitrary number of controllable languages is controllable.

Definition 4 *Assuming that $\Sigma_c \subseteq \Sigma_m$, a prefix-closed language K is observable w.r.t. $L(G)$ and Σ_m if $P^{-1}_{\Sigma_m}[P_{\Sigma_m}(K)] \cap L(G) \subseteq K$*[3]. •

Intuitively, K is observable, if K can be exactly recovered from its projection $P_{\Sigma_m}(K)$ and $L(G)$. Note that this is a necessary condition for a controller that forces the system to behave like K to be implementable. In other words, from a control point of view, when disabling an event c after the execution of s, then c has to be disabled after all execution traces of $[s]_m$. Under the assumption $\Sigma_c \subseteq \Sigma_m$, the union of an arbitrary number of observable languages is observable. Therefore, under this assumption, both controllability and observability are stable under union of languages, and there exists a supremal controllable and observable prefix-closed sub-language of K, that we denote

$$\text{CO}^{\uparrow}(K, L(G), \Sigma_c, \Sigma_m) \tag{3}$$

[3]Note that we have given here the formal definition of normality. Under the assumption $\Sigma_c \subseteq \Sigma_m$, observability and normality coincide [3].

The language $\mathrm{CO}^{\uparrow}(K, L(G), \Sigma_c, \Sigma_m)$ represents the largest behavior included in $K(\subseteq L(G))$ that can be enforced by control. Moreover, $\mathrm{CO}^{\uparrow}$ is monotone in the first argument.

Lemma 3 *Assuming that $\Sigma_c \subseteq \Sigma_m$, let $s \in K \setminus \mathrm{CO}^{\uparrow}(K, L(G), \Sigma_c, \Sigma_m)$, then*

$$[s]_m \cap \mathrm{CO}^{\uparrow}(K, L(G), \Sigma_c, \Sigma_m) = \emptyset$$

Proof Because $\mathrm{CO}^{\uparrow}(K, L(G), \Sigma_c, \Sigma_m)$ is observable, this set and its relative complement are unions of equivalence classes of $\sim_m$. ◇

Similarly to lemma 2, the equivalence classes of $L(G)$ w.r.t. Σ_m are preserved by control.

5 Enforcing opacity by control

Our purpose is to solve the opacity control problem stated as follows.

Problem: Show that the set of controllable and observable restrictions (i.e. sub-languages) of $L(G)$ enforcing the opacity of L_φ either is empty or has a greatest element and compute this maximal permissive controllable and observable sub-language of $L(G)$.

In the sequel, we shall assume that an attacker has a full knowledge of the structure of G, knows the interface of the controller Σ_m and is able to perform in his head all calculations that the administrator has made to compute this controller. In particular, this entails that the structure of the controlled system may be available to the attacker, thus possibly inducing new confidential information flow. This assumptions are at present informal, but might be formalized e.g. using language theory and epistemic logic. Moreover, *in the rest of the paper, it is always assumed that $\Sigma_c \subseteq \Sigma_m$ (the controllable events are observed by the controller).*

5.1 Characterization of the solution

We now investigate the existence of a supremal solution to the opacity control problem. To do so, we consider the set

$$\mathcal{C}_\varphi = \{L \subseteq L(G) \mid \begin{array}{l} L_\varphi \text{ is opaque w.r.t. } L \text{ and } \Sigma_a, \\ L \text{ is prefix-closed}, \\ L \text{ is controllable w.r.t. } L(G) \text{ and } \Sigma_c, \\ L \text{ is observable w.r.t. } L(G) \text{ and } \Sigma_m\} \end{array}$$

and the prefix-closed language

$$\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c) = \bigcup_{L \in \mathcal{C}_\varphi} L \tag{4}$$

Proposition 2 *If $\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c) \neq \emptyset$, then it is the supremal sub-language of $L(G)$ such that*

(1) *$\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ is controllable and observable w.r.t. $L(G)$, Σ_c and Σ_m,*

(2) *and L_φ is opaque w.r.t. $\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ and Σ_a.*

Otherwise, no control can enforce the opacity of L_φ.

Proof If $\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c) \neq \emptyset$, then it is the union of an arbitrary number of languages that are controllable, observable and such that L_φ is opaque w.r.t. the corresponding restrictions of $L(G)$. These three properties are stable under arbitrary union of languages (under the hypothesis that $\Sigma_c \subseteq \Sigma_m$). So $\text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ satisfies (1) and (2). ◇

Even though the previous proposition entails the existence of a unique maximal sub-language of $L(G)$, that is controllable, observable and in restriction to which L_φ is opaque, we still have to examine whether this language is regular (or at least, to exhibit sufficient conditions for regularity) and to provide an effective computation of this language.

It may be remarked that restricting languages to ensure controllability and observability does not always preserve opacity and the other way round (See Example 2). Thus, in a first attempt towards an effective computation of CO-OP$^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$, following the classical methodology of Supervisory Control Theory[4], we establish below a fix-point characterization of this language by alternating the computation of the supremal sub-language that ensures the opacity of L_φ and the supremal controllable and observable sub-language.

Consider the operator

$$\mathcal{K}(\bullet) = \text{CO}^\uparrow(\text{OP}^\uparrow(\bullet, L_\varphi, \Sigma_a), L(G), \Sigma_c, \Sigma_m).$$

Remark that $\mathcal{K}(\bullet)$ is monotone w.r.t. set inclusion. Now, as the prefix-closed subsets of $L(G)$ form a complete sub-lattice of $\mathcal{P}(\Sigma^*)$, it follows from Knaster-Tarski's Theorem [10] that $\mathcal{K}(\bullet)$ has a greatest fix-point in this sub-lattice. Let $K(L(G), L_\varphi)$ be the greatest fix-point of the operator $\mathcal{K}(\bullet)$ included in $L(G)$[5].

Proposition 3 $K(L(G), L_\varphi) = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$

Proof We denote $L^c = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$. Clearly, L_φ is opaque w.r.t. $K(L(G), L_\varphi)$ and Σ_a. This language is controllable and observable, hence $K(L(G), L_\varphi) \subseteq L^c$.

Moreover, we have $L^c \subseteq L(G) = \mathcal{K}^0(L(G))$. Assume now that $L^c \subseteq \mathcal{K}^i(L(G))$ for some i. Then, from the monotony of $\mathcal{K}(\bullet)$, we get $\mathcal{K}^{i+1}(L(G)) \supseteq \mathcal{K}(L^c) = L^c$, since L^c controllable and observable and L_φ is opaque w.r.t. L^c and Σ_a. By transfinite induction, it follows that $L^c \subseteq \mathcal{K}^\alpha(L(G))$ for every ordinal α. Therefore $L^c \subseteq \cap_\alpha \mathcal{K}^\alpha(L(G)) = K(L(G), L_\varphi)$. ◇

Note that this fix-point characterization of CO-OP$^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ does not ensure that this language can be always computed by a finite iteration as the following example shows.

Example 2 *Consider the LTS G shown in Fig. 5 where* $\Sigma_a = \{A, B, c\}$, $\Sigma_m = \Sigma$, $\Sigma_c = \{c\}$ *and the predicate* L_φ *is the set of the sequences that reach the states represented with squares in G. Let* $K_i = \mathcal{K}^i(L(G))$ *denote the language computed after i iterations of the operator* $\mathcal{K}(\bullet)$.

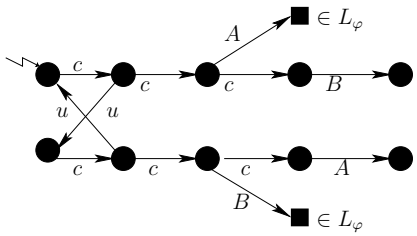


Figure 5: $L(G)$ and L_φ

In $L(G)$*, the sole string that belongs to* L_φ*, and therefore reveals it, is c.c.A, which requires to disable the second event c, seeing that A is uncontrollable. The LTS that generates* K_1 *is represented in Fig. 6(a).*

In K_1*, c.c.A has disappeared and the sole string that belongs to* L_φ*, and therefore reveals it, is is c.u.c.c.B, which requires to disable the event c after c.u.c. The result (*K_2*) is depicted in Fig. 6(b). After* $2i$ *iterations of the operator* $\mathcal{K}(\bullet)$*, one gets the language* K_{2i} *generated by the LTS depicted in Fig. 7(a).*

[4]that ensures both non-blocking and controllability

[5]$K(L(G), L_\varphi)$ is also the greatest fix-point of the operator $\mathcal{K}' = \text{OP}^\uparrow(\text{CO}^\uparrow(\bullet, L(G), \Sigma_c, \Sigma_m), L_\varphi, \Sigma_a)$.

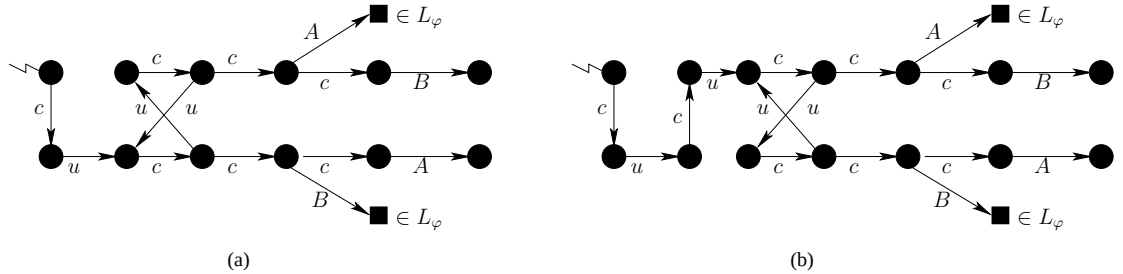


Figure 6: K_1 and K_2

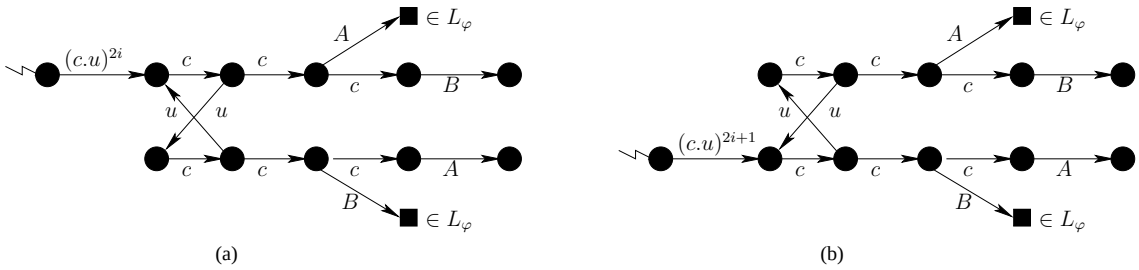


Figure 7: K_{2i} and K_{2i+1}

In K_{2i}, the string $(c.u)^{2i}.c.c.A$ reveals L_φ and it must be eliminated by disabling the last c, which is done in the language K_{2i+1} (See Fig. 7(b). But, in K_{2i+1}, the string $(c.u)^{2i+1}.c.c.B$ reveals L_φ. Disabling the last c leads to K_{2i+2}, which reproduces the situation found in K_{2i} (up to replacing the prefix $(c.u)^{2i}$ by $(c.u)^{2i+2}$. Finally, even though the limit $K(L(G), L_\varphi)$ of this decreasing chain is the regular language $(c.u.c.u)^*$, the fix-point iteration produces a strictly decreasing and infinite sequence of languages K_j showing that the above algorithm may not terminate. ◇

In the rest of the paper, we investigate sufficient conditions, induced by relations between the alphabets Σ_c, Σ_a and Σ_m, under which CO-OP$^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ is regular and one can effectively compute uniformly from the arguments of CO-OP$^\uparrow(\bullet)$ a finite automaton generating this optimal opacity control.

But first, we establish a proposition that helps to simplify the remaining proofs (In Sections 5.2.2 and 5.2.3). This proposition states that whenever $\Sigma_a \subseteq \Sigma_m \subseteq \Sigma$, we can reformulate the control problem in terms of the observed system w.r.t. Σ_m and a new predicate $L^m_\varphi \subseteq \Sigma^*_m$ derived from L_φ and Σ_m, solve the problem in this new setting (thus, under full observation) and lift up the solution to the initial setting. The intuition is that observing events from $\Sigma \setminus \Sigma_m$ could not help the controller anyway.

Proposition 4 *Assume that $\Sigma_a \subseteq \Sigma_m \subseteq \Sigma$ and $\Sigma_c \subseteq \Sigma_m$. Let L_φ, $L(G) \subseteq \Sigma^*$, then*

$$\text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c) = P^{-1}_{\Sigma_m}(\text{CO-OP}^\uparrow(P_{\Sigma_m}(L(G)), L^m_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)) \cap L(G)$$

where $L^m_\varphi = \{\rho \in P_{\Sigma_m}(L(G)) : P^{-1}_{\Sigma_m}(\rho) \cap L(G) \subseteq L_\varphi\}$

Proof Consider the following languages:

- $K = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$,
- $F = \text{CO-OP}^\uparrow(P_{\Sigma_m}(L(G)), L^m_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$
- $H = P^{-1}_{\Sigma_m}(F) \cap L(G)$.

We will prove that $H = K$.

Let us first prove that L_φ is opaque w.r.t. H and Σ_a. Consider $s \in H \cap L_\varphi$. As $P_{\Sigma_m}(s) \in F$ and L_φ^m is opaque w.r.t. F and Σ_a (by definition of F), there exists $\rho \in F$ such that $\rho \sim_a P_{\Sigma_m}(s)$ and $\rho \in F \setminus L_\varphi^m$. Then, $\exists s' \in P_{\Sigma_m}^{-1}(\rho) \cap L(G), s' \notin L_\varphi$ according to the definition of L_φ^m. But $\rho \in F$ implies that $s' \in H$. $P_{\Sigma_m}(s') \sim_a P_{\Sigma_m}(s)$ and $\Sigma_a \subseteq \Sigma_m$ implies that $s' \sim_a s$. So L_φ is opaque w.r.t. H and Σ_a.

Let us now show that H is controllable. Consider $s \in H, \sigma \in \Sigma \setminus \Sigma_c$ such that $s\sigma \in L(G)$. Let $\rho = P_{\Sigma_m}(s)$. By definition of H, we get $\rho \in F$.

- If $\sigma \notin \Sigma_m$ then $P_{\Sigma_m}(s\sigma) = \rho$ and finally $s\sigma \in H$.
- If $\sigma \in \Sigma_m$, we have $\rho \in F$ and $\rho\sigma \in P_{\Sigma_m}(L(G))$. As F is controllable, we get $\rho\sigma \in F$, which entails $s\sigma \in H$ as $s\sigma \in P_{\Sigma_m}^{-1}(\{\rho\sigma\})$.

Finally we note that H is observable by construction. As K is the supremal controllable and observable sub-language of $L(G)$ for which L_φ is opaque, we can conclude that $H \subseteq K$.

Let us now prove that $P_{\Sigma_m}(K) \subseteq F$.

- Let $\rho \in P_{\Sigma_m}(K)$. There exists $s \in K$ such that $P_{\Sigma_m}(s) = \rho$. Since L_φ is opaque w.r.t. K and Σ_a, there exists $s' \in K, s' \sim_a s$ such that $s' \in K \setminus L_\varphi$. Let $\rho' = P_{\Sigma_m}(s')$. We have $\rho' \notin L_\varphi^m$. As $\rho \sim_a \rho'$, we conclude that L_φ^m is opaque w.r.t. $P_{\Sigma_m}(K)$ and Σ_a.
- Let us show that $P_{\Sigma_m}(K)$ is controllable w.r.t. $P_{\Sigma_m}(L(G))$ and Σ_c. Let $\rho \in P_{\Sigma_m}(K)$ and $\sigma \in \Sigma_m \setminus \Sigma_c$ such that $\rho\sigma \in P_{\Sigma_m}(L(G))$. Then, $\exists s \in KK \subseteq L(G)$, such that $P_{\Sigma_m}(s) = \rho$ and $s\sigma \in K(\Sigma_m \setminus \Sigma_c) \cap L(G)$. Since K is controllable, $s\sigma \in K$ and then $\rho\sigma = P_{\Sigma_m}(s\sigma) \in P_{\Sigma_m}(K)$. So $P_{\Sigma_m}(K)$ is controllable.

Now, $P_{\Sigma_m}(K)$ is obviously observable and we get that $P_{\Sigma_m}(K) \subseteq F$. This implies that $P_{\Sigma_m}^{-1}(P_{\Sigma_m}(K)) \cap L(G) \subseteq H$ and since $K = K \cap L(G) \subseteq P_{\Sigma_m}^{-1}(P_{\Sigma_m}(K)) \cap L(G)$, we conclude that $K \subseteq H$ and finally that $H = K$. ◇

5.2 Effective computation of the supremal solution

Next, we investigate three sufficient conditions under which CO-OP$^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ is regular and effectively computable.These conditions bear upon the inclusion relationships between the alphabets Σ_a, Σ_m and Σ_c.

5.2.1 Assumption 1: $\Sigma_c \subseteq \Sigma_m \subseteq \Sigma_a \subseteq \Sigma$

Under this assumption, the controller observes and controls only a part of the actions of the attacker, meaning that it is less powerful than the attacker. Nevertheless, this is a sufficient condition allowing to solve the control problem.

Proposition 5 *Assume $\Sigma_c \subseteq \Sigma_m \subseteq \Sigma_a \subseteq \Sigma$, then K_1 $(= \mathcal{K}(L(G)))$ $=$ CO-OP$^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ is regular and effectively computable.*

Proof Let $L_1 = \text{OP}^\uparrow(L(G), L_\varphi, \Sigma_a)$, then $K_1 = \text{CO}^\uparrow(L_1, L(G), \Sigma_c, \Sigma_m)$. Consider $s \in K_1 \cap L_\varphi$. As L_φ is opaque w.r.t. L_1 and Σ_a, $\exists s' \in L_1$ such that $s \sim_a s'$ and $s' \notin L_\varphi$. As $\Sigma_m \subseteq \Sigma_a$ and $s \sim_a s'$, we get $s \sim_m s'$. Hence, as an immediate consequence of Lemma 3, we also have $s' \in K_1$, which entails that L_φ is opaque w.r.t. K_1 and Σ_a. Hence, $K_1 = K(L(G), L_\varphi)$, which according to Proposition 3 entails that $K_1 = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$. ◇

5.2.2 Assumption 2: $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_m \subseteq \Sigma$

This assumption simply means that the controller can observe all the actions of the attacker and control them.

Based on proposition 4, one can assume, without loss of generality, that $\Sigma_m = \Sigma$.

Proposition 6 *Assume* $\Sigma_a \subseteq \Sigma_c \subseteq \Sigma_m = \Sigma$, *then* K_1 $(= \mathcal{K}(L(G)))$ $= \text{CO-OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a, \Sigma_m, \Sigma_c)$ *is regular and effectively computable.*

Proof We first show that $\text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a)$ is controllable with respect to L, Σ_c. Consider $s \in \text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a)$ and $\sigma \notin \Sigma_c$, such that $s\sigma \in L(G)$. As $\Sigma_a \subseteq \Sigma_c$, $\sigma \notin \Sigma_a$ and then $s\sigma \in [s]_a \cap L(G)$ and according to Lemma 2, $s\sigma \in \text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a)$, which is then controllable w.r.t. L, Σ_c and observable w.r.t. Σ_c and Σ_m since $\Sigma_m = \Sigma$. Hence,

$$\text{CO}^{\uparrow}(\text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a), L(G), \Sigma_c, \Sigma_m) = \text{OP}^{\uparrow}(L(G), L_\varphi, \Sigma_a) = K(L(G), L_\varphi)$$

and we conclude using the result of Proposition 3. ◇

5.2.3 Assumption 3: $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_m \subseteq \Sigma$

Under this assumption, even though all actions of the attacker can be observed by the controller, only a part of them can be controlled. One can think that the controller can filter out the requests sent by the attacker to the system, whereas the outputs of the system cannot be disabled by the controller. This is for example the behavior of a firewall for Internet services.

It is easy to check that the system of Example 2, for which the fix-point computation does not terminate, fulfills the assumption of this subsection. This leads us to design a new algorithm.

Using proposition 4, we can assume that $\Sigma_m = \Sigma$. We also make the following assumption without loss of generality. The system is given by a deterministic LTS $G = (Q_G, \Sigma, q_0^G, \delta_G)$. The predicate L_φ is specified by a complete and deterministic LTS $S_\varphi = (Q_S, \Sigma, q_0^S, \delta_S)$ with a set F_φ of final states such that $L_\varphi = L_{F_\varphi}(S_\varphi)$ and $L(S_\varphi) = \Sigma^*$.

First, we perform the product of G and S_φ in order to tag the states in which the predicate L_φ is satisfied: $G_\varphi = G \parallel S_\varphi = (Q, \Sigma, q_0, \delta)$, with $Q = Q_G \times Q_S$, $q_0 = (q_0^G, q_0^S)$ and δ the synchronized transition function. By denoting $F = Q_G \times F_\varphi$, we get $L_F(G_\varphi) = L(G) \cap L_\varphi$, meaning that the execution traces that reach or go through a state of F reveal L_φ (note that $L(G_\varphi) = L(G)$, because S_φ is complete). Thus, L_φ is opaque w.r.t. $L(G_\varphi) \Leftrightarrow L_\varphi$ is opaque w.r.t. $L(G)$. Clearly, if L_φ is non-opaque w.r.t. $L(G_\varphi) \setminus \Sigma^*\Sigma_c\Sigma^*$ and Σ_a, then no control can enforce the opacity of L_φ. So in the sequel, without loss of generality, we assume that L_φ is opaque w.r.t. $L(G_\varphi) \setminus \Sigma^*\Sigma_c\Sigma^*$ and Σ_a. In particular, this entails that L_φ is opaque w.r.t. $L(G_\varphi) \setminus \Sigma_c^*$.

Under this assumption, we show that the optimal opacity control may be enforced by a finite state controller, defined by a *deterministic* LTS $C = (\mathcal{Q}, \Sigma, \Theta_0, \delta)$ with the set of states

$$\mathcal{Q} = \{(X, q) : q \in X \subseteq Q\}$$

and the initial state $\Theta_0 = (X_0, q_0)$ specified by

$$X_0 = \{q \in Q : \exists s \in (\Sigma \setminus \Sigma_a)^*,\ q_0 \xrightarrow{s} q\}$$

Intuitively, after the execution of a trace s, the controller is in a state (X, q) when the controlled system is in state q (recall that $\Sigma_m = \Sigma$) and X is the best estimate of the current state of G_φ that the attacker $\mathcal{A}$ can get from the observation $P_{\Sigma_a}(s)$ of this execution trace. In particular, if no event in Σ_a has been produced yet, the best estimate is X_0 (recall that the attacker has full knowledge of the structure of G_φ)[6].

[6]Some states in $\mathcal{Q}$ will possibly be not reachable, but it does not matter since these states can be eliminated afterwards by trimming C.

In the sequel, we denote $\mathcal{T}$ the set of transitions of G_φ.

The main task, for completing the construction of C, is to determine the map $\alpha : 2^Q \longrightarrow 2^{\mathcal{T}}$ that tells, for each state (X, q) and simultaneously for all $q \in X$, which set $\alpha(X)$ of controllable transitions of G_φ the controller does enable, thus

$$\alpha(X) \subseteq \alpha_0(X) \stackrel{\Delta}{=} \{q \stackrel{\sigma}{\to} q' \in \mathcal{T} : q \in X, q' \in Q, \sigma \in \Sigma_c\}.$$

So, in state (X, q), the controller disables the transitions $q \stackrel{\sigma}{\to} q' \in \alpha_0(X) \setminus \alpha(X)$, all of which are controllable.

Suppose the correct map α has been computed. Then the set $\mathcal{T}_C$ (of transitions of C) is inductively defined as the least set of transitions $(X, q) \stackrel{\sigma}{\to}_C (X', q')$ such that (X, q) is reachable, $q \stackrel{\sigma}{\to} q'$, and the estimate X' of the attacker is updated from X as follows:

- if $\sigma \notin \Sigma_a$, then $X' = X$
- if $\sigma \in (\Sigma_a \setminus \Sigma_c)$, then

$$X' = \{\ \overline{q}' \in Q :\ \exists \overline{q} \in X,\ \exists s \in \Sigma^*, \overline{q} \stackrel{s}{\to} \overline{q}' \text{ and } \sigma \sim_a s\} \tag{5}$$

- if $\sigma \in \Sigma_c$, then

$$X' = \ \{\overline{q}" \in Q :\ \exists \overline{q} \in X, \exists \overline{q}' \in Q,\ \exists s \in \Sigma^*, \overline{q} \stackrel{\sigma}{\to} \overline{q}' \in \alpha(X) \text{ and } \overline{q}' \stackrel{s}{\to} \overline{q}" \text{ and } s \sim_a \epsilon\} \tag{6}$$

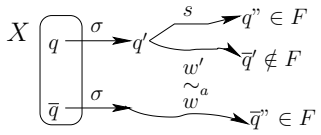
This is coherent with the idea that the attacker has full knowledge of the structure of C, hence of α.

Lemma 4 *Let $(X, q) \stackrel{\sigma s}{\to}_C (X', q')$ with $\sigma \in \Sigma_a$ and $s \in (\Sigma \setminus \Sigma_a)^*$, then $\forall \overline{q'} \in X'$, $\exists \overline{q} \in X$, $(X, \overline{q}) \stackrel{w}{\to}_C (X', \overline{q}')$ with $w \sim_a \sigma$.* ◇

Lemma 5 *Let $\Theta_0 \stackrel{s}{\to}_C (X, q)$ and $\Theta_0 \stackrel{s'}{\to}_C (X', q')$. If $s \sim_a s'$ then $X = X'$.* ◇

Both lemmas are immediate consequences of the definition of $\mathcal{T}_C$.

We explain now the motivation under the definition of the map α. Let (X, q) be a reachable state of the controller, thus $q \in X$, and let $q \stackrel{\sigma}{\to} q' \in \mathcal{T}$ with $\sigma \in \Sigma_c$. If, for some $s \in (\Sigma \setminus \Sigma_c)^*$, $q' \stackrel{s}{\to} q" \in F$ but $\sigma s \sim_a s'$ for no sequence s' such that $\overline{q} \stackrel{s'}{\to} \overline{q}" \notin F$ for some $\overline{q} \in X$, then the controller C should disable $q \stackrel{\sigma}{\to} q'$ when in state (X, q), otherwise triggering s after σ will reveal L_φ. Hence, one should have $q \stackrel{\sigma}{\to} q' \notin \alpha(X)$.



But now suppose that, for some $w, w' \in (\Sigma \setminus \Sigma_c)^*$ and $\overline{q} \in X$, $\overline{q}'$, $\overline{q}" \in Q$, $\overline{q} \stackrel{\sigma w}{\to} \overline{q}" \in F$, $w \sim_a w'$, and $q' \stackrel{w'}{\to} \overline{q}' \notin F$ (thus $\neg(s \sim_a w')$). If $q \stackrel{\sigma}{\to} q' \notin \alpha(X)$, and the attacker has full knowledge of C and the map α, the transition sequence $\overline{q} \stackrel{\sigma w}{\to} \overline{q}"$ may now reveal the predicate L_φ, since the attacker knows that the masking transition sequence $q \stackrel{\sigma w'}{\to} \overline{q}'$ is disabled by C. Therefore, $\alpha(X)$ must be computed iteratively as the limit of a decreasing chain started from the finite set $\alpha_0(X)$.

The definition of $\alpha(X)$ is as follows. Let T range over the subsets of $\alpha_o(X)$, and for $\sigma \in \Sigma_c$, let

$$Next(X, \sigma, T) \stackrel{\Delta}{=} \{q' \in Q : \exists q \in X, q \stackrel{\sigma}{\to} q' \in T\}$$

then

$$\alpha(X) \triangleq gfp(\lambda T.\alpha_0(X) \cap Accept(X,T)),$$

where $Accept(X,T) = T \setminus Bad(X,T)$ letting

$$Bad(X,T) \triangleq \{t \in T \ : \ t = q \xrightarrow{\sigma} q',\ q \in X,\ \sigma \in \Sigma_c, \\ P_{\Sigma_a}(L_{\{q'\},F}(G) \cap (\Sigma \setminus \Sigma_c)^*) \setminus P_{\Sigma_a}(L_{Next(X,\sigma,T),(Q\setminus F)}(G)) \neq \emptyset\}$$

All transitions in $Bad(X,T)$ should be disabled by control, because they may lead to a confidential information flow by triggering one controllable event followed by an uncontrollable sequence of events.

Remark 2 *The controller C is computed by independent iterations of the operator $Accept(X,T)$ for all $X \subseteq Q$ and $T \subseteq \alpha_0(X)$.* •

We will now prove that L_φ is opaque w.r.t. $L(C)$, that $L(C)$ is controllable and observable, and that it is the supremal sub-language of $L(G) = L(G_\varphi)$ with these properties.

Proposition 7 *L_φ is opaque w.r.t. $L(C)$ and Σ_a*

Proof Consider $s \in L(C) \cap L_\varphi \subseteq L(G)$.

- If $P_{\Sigma_c}(s) = \varepsilon$, then $s \notin \Sigma^*\Sigma_c\Sigma^*$. As by hypothesis L_φ is opaque w.r.t. $L(G_\varphi) \setminus (\Sigma^*\Sigma_c\Sigma^*)$ and Σ_a, there exists $s' \in L(G_\varphi)$ such that $s' \sim_a s$ and $s' \notin L_\varphi$. Since $\Sigma_c \subseteq \Sigma_a$, we also have $P_{\Sigma_c}(s') = \varepsilon$ and hence $s' \in L(C)$.
- if $P_{\Sigma_c}(s) \neq \varepsilon$, then s can be decomposed as $s = s_1 c s_2$ with $c \in \Sigma_c$ and $s_2 \in (\Sigma \setminus \Sigma_c)^*$. There exists $(X,q) \in \mathcal{Q}$ and $q_1 \in Q$ such that $\Theta_0 \xrightarrow{s_1} (X,q)$ and $(q \xrightarrow{c} q_1) \in \alpha(X)$. Let $(X,q) \xrightarrow{c}_C (X',q_1)$, thus
$$X' = \{q" \ : \ \exists q' \in Next(X,c,\alpha(X)), \exists s \in \Sigma^*, q' \xrightarrow{s} q" \text{ and } s \sim_a \epsilon\}$$
Assume for a contradiction that $\forall q'_1 \in X'$, $\forall s'_2 \in [s_2]_a$, $(X',q'_1) \xrightarrow{s'_2}_C (Z,q'_2)$ entails $q'_2 \in F$. Then $s'_2 \in L_{\{q'_1\},F}(G_\varphi) \cap (\Sigma \setminus \Sigma_c)^*$ and $P_{\Sigma_a}(s_2) \notin P_{\Sigma_a}(L_{X',(Q\setminus F)}(G_\varphi))$, hence $(q \xrightarrow{c} q_1) \in Bad(X,\alpha(X))$, in contradiction with $(X,q) \xrightarrow{c}_C (X',q_1)$. Therefore, $(X',q'_1) \xrightarrow{s'_2}_C (Z,q'_2)$ for some $q'_1 \in X'$, $s'_2 \in [s_2]_a$, and $q'_2 \notin F$. Now, $(X,q) \xrightarrow{c}_C (X',q_1)$ and $q'_1 \in X'$ entail that $(X,\overline{q}) \xrightarrow{cs'}_C (X',q'_1)$ for some $\overline{q} \in X$ and $s' \in (\Sigma \setminus \Sigma_a)^*$ (Lemma 4), and $\overline{q} \in X$ entails that $\Theta_0 \xrightarrow{s'_1}_C (X,\overline{q})$ for some $s'_1 \sim_a s_1$ (Lemma 4). Altogether, $s = s_1 c s_2 \sim_a s'_1 c s' s'_2 \in L(C) \setminus L_\varphi$. So L_φ is opaque w.r.t. $L(C)$ and Σ_a. ◇

Proposition 8 *$L(C)$ is controllable w.r.t. $L(G_\varphi)$ and Σ_c.*

Proof Let $s \in L(C)$ and $\sigma \in \Sigma \setminus \Sigma_c$ such that $s\sigma \in L(G_\varphi)$. Then, $\exists (X,q) \in \mathcal{Q}$ such that $\Theta_0 \xrightarrow{s} (X,q)$ in C and $\exists q' \in Q$ such that $q \xrightarrow{\sigma} q'$ in G_φ. Since $\sigma \notin \Sigma_c$, $(X,q) \xrightarrow{c}_C (X'q')$ for some q' by definition of $\mathcal{T}_C$, hence $s\sigma \in L(C)$. ◇

Theorem 1 *Assume that $\Sigma_c \subseteq \Sigma_a \subseteq \Sigma_m = \Sigma$, then $L(C) = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma, \Sigma_c)$.*

Proof Let $K = \text{CO-OP}^\uparrow(L(G), L_\varphi, \Sigma_a, \Sigma, \Sigma_c)$. Since L_φ is opaque w.r.t. $L(C)$ and Σ_a, $L(C)$ is controllable w.r.t. $L(G_\varphi)$ and Σ_c, and $L(C)$ is observable w.r.t. Σ_m and $L(G)$ (as $\Sigma_m = \Sigma$), we have $L(C) \subseteq K$.

It remains to prove that $K \subseteq L(C)$. We proceed by contradiction. Let $s \in K \setminus L(C)$. This sequence can be decomposed as $s = s_1 \sigma s_2$, where s_1 is the *longest prefix* of s such that $[s_1]_a \cap K = [s_1]_a \cap L(C)$, hence $s_1 \in L(C)$ (because $s_1 \in K$), $[s_1\sigma]_a \cap K \neq [s_1\sigma]_a \cap L(C)$ (by definition of s_1), and $\sigma \in \Sigma_a$ (because $[s_1\sigma]_a \neq [s_1]_a$).

Since $L(C) \subseteq K$, $[s_1\sigma]_a \cap L(C) \subset [s_1\sigma]_a \cap K$ and one can find $u = u_1\sigma u_2$ in $[s_1\sigma]_a \cap K$, with $u_1 \sim_a s_1$ and $u_2 \sim_a \epsilon$, such that $u_1\sigma u_2 \notin L(C)$. As $s_1 \in L(C) \cap K$, $u_1 \sim_a s_1$ and $u_1 \in K$, necessarily $u_1 \in L(C)$ by definition of s_1. As $u_1 \in L(C)$, $u_2 \in (\Sigma \setminus \Sigma_c)^*$, $u_1\sigma u_2 \in L(G_\varphi) \setminus L(C)$, and $L(C)$ is controllable, necessarily $u_1\sigma \notin L(C)$ and $\sigma \in \Sigma_c$.

Since $u_1 \in L(C)$, there exists $(X, q) \in \mathcal{Q}$ such that $\Theta_0 \xrightarrow{u_1} (X, q)$. By construction of C, $q_0 \xrightarrow{u_1} q$ in G_φ, $q \in X$, and $X = \{q_1 \in Q : \exists u \in [u_1]_a \cap L(C), q_0 \xrightarrow{u} q_1\}$. It also follows from the construction of C that

$$\alpha(X) = \{q_1 \xrightarrow{c} q_2 : q_1 \in X,\ q_2 \in Q,\ c \in \Sigma_c \text{ s.t. } \exists u \in [u_1]_a \cap L(C), q_0 \xrightarrow{u} q_1 \text{ and } uc \in L(C)\}$$

and moreover, for all $q_1 \xrightarrow{c} q_2 \in \alpha(X)$, $\forall u \in [u_1]_a \cap L(C)$, $q_0 \xrightarrow{u} q_1 \Rightarrow uc \in L(C)$. Therefore, $u_1 \in [u_1]_a \cap L(C)$, $q_0 \xrightarrow{u_1} q, \sigma \in \Sigma_c$, and $u_1\sigma \notin L(C)$ entail that $\alpha(X)$ contains no transition $q \xrightarrow{\sigma} q'$.

Consider now the alternative set of transitions

$$\beta = \{q_1 \xrightarrow{c} q_2 \ : q_1 \in X, q_2 \in Q, \sigma \in \Sigma_c \text{ s.t. } \exists u \in [u_1]_a \cap K, q_0 \xrightarrow{u} q_1 \text{ and } u\sigma \in K\}$$

Then clearly $\alpha(X) \subseteq \beta$ (because $L(C) \subseteq K$) and β contains a transition $q \xrightarrow{\sigma} q'$ (because $q_0 \xrightarrow{u_1} q$, $u_1\sigma \in K$, and $K \subseteq L(G_\varphi)$). Therefore, $\alpha(X) \subset \beta$.

In order to complete the proof, we will show that $(q_1 \xrightarrow{c} q_2) \notin Bad(X, \beta)$ for all $c \in \Sigma_c$ and $(q_1 \xrightarrow{c} q_2) \in \beta$, entailing that $Accept(X, \beta) = \beta \setminus Bad(X, \beta) = \beta$, and hence that $\beta \subseteq \alpha(X)$ in view of the greatest fixpoint definition of $\alpha(X)$, resulting in a contradiction with $\alpha(X) \subset \beta$.

Let $(q_1 \xrightarrow{c} q_2) \in Bad(X, \beta)$. Recalling that $Bad(X, \beta) \subseteq \beta$ and that $[u_1]_a = [s_1]_a$, let $u \in [u_1]_a \cap L(C) = [u_1]_a \cap K$ such that $\Theta_0 \xrightarrow{u}_C (X, q_1)$ and $uc \in K$. As $(q_1 \xrightarrow{c} q_2) \in Bad(X, \beta)$, there must exist $v \in (\Sigma \setminus \Sigma_c)^*$ such that $ucv \in L_\varphi \cap L(G_\varphi)$. K is controllable, hence $ucv \in K$. L_φ is opaque w.r.t. K and Σ_a, then $\exists w \in [ucv]_a \cap K$ such that $w \notin L_\varphi$. As $w \in [ucv]_a$ and $c \in \Sigma_c \subseteq \Sigma_a$, there should exist $w_1, w_2 \in \Sigma^*$ such that $w = w_1 c w_2$ with $w_1 \sim_a u \sim_a u_1$ and $w_2 \sim_a v$. Now, $w_1 \in [u_1]_a \cap K$ $\Rightarrow w_1 \in [u_1]_a \cap L(C) \Rightarrow \exists q_3 \in X$, $\Theta_0 \xrightarrow{w_1} (X, q_3)$ in C, by Lemma 5. As $w_1 c \in K$, there must exist $q_4 \in Q$ such that $(q_3 \xrightarrow{c} q_4) \in \beta$ and thus $q_4 \in Next(X, c, \beta)$. Now, $v \in L_{\{q_2\},F}(G) \cap (\Sigma \setminus \Sigma_c)^*$, $w_2 \in L_{\{q_4\},(Q \setminus F)}(G) \cap (\Sigma \setminus \Sigma_c)^*$, and $v \sim_a w_2$, so based on the definition of the Bad operator, $(q_1 \xrightarrow{c} q_2) \notin Bad(X, \beta)$, which is the expected contradiction. ◇

Example 3 *To illustrate the algorithm, let us come back to our previous example, where* $F = \{3, 11\}$.

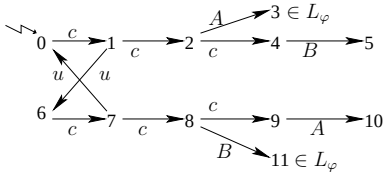


Figure 8: $L(G)$ and L_φ

At the first step of the computation of $L(C)$, *we get* $X_0 = \{0\}$, $\Theta_0 = (X_0, 0)$*) and* $\alpha_0(X_0) = \{(0 \xrightarrow{c} 1)\}$. *Now, we also have* $P_{\Sigma_a}(L(G_\varphi, 1, F) \cap (\Sigma \setminus \Sigma_c)^*) = \emptyset$, *implying* $(0 \xrightarrow{c} 1) \notin Bad(X_0, \alpha_0(X_0))$, *and thus* $\alpha(X_0) = \{(0 \xrightarrow{c} 1)\}$. *Thus, in C, we have* $\Theta_0 \xrightarrow{c}_C (\{1, 6\}, 1)$.
Further, for $X_1 = \{1, 6\}$, *we get* $\alpha_0(X_1) = \{(1 \xrightarrow{c} 2), (6 \xrightarrow{c} 7)\}$, $Next(X_1, c, \alpha_0(X_1)) = \{2, 7\}$,

$$P_{\Sigma_a}(L(G_\varphi, 2, F) \cap (\Sigma \setminus \Sigma_c)^*) = \{A\}$$

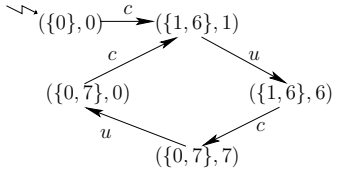


Figure 9: The corresponding supervisor C

and $P_{\Sigma_a}(L(G_\varphi, Next(X_1, c, \alpha_0(X_1)), Q \setminus F) = (cc)^+.(A + cB) + \overline{cB} + cc$

Thus, $1 \xrightarrow{c} 2 \in Bad(X_1, \alpha_o(X_1))$ *and* $(1 \xrightarrow{c} 2) \notin \alpha_1(X_1)$*. Finally, it can be shown that* $\{(6 \xrightarrow{c} 7)\} = \alpha(X_1)$*. The other values of* $\alpha(X)$ *for* $X \in \mathcal{Q}$ *can be computed the same way. The resulting* C *is given by the LTS depicted in Figure 9* $\diamond$

Acknowledgment: The authors would like to thank the reviewers and Thierry Jéron for their helpful comments.

6 Conclusion

Given a finite transition system G over Σ and a regular predicate $L_\varphi \subseteq \Sigma^*$, we have addressed the problem of computing a supervisor C that enforces the opacity of L_φ against an attacker with alphabet $\Sigma_a \subseteq \Sigma$, supposedly trying to push $G \times C$ to reveal L_φ (i.e. to produce an execution s such that $P_{\Sigma_a}(s) = P_{\Sigma_a}(s') \Rightarrow s' \in L_\varphi$ for all $s' \in L(G \times C)$). We have shown how computing the optimal finite state supervisor C with controllable (observable) alphabet Σ_c (Σ_m) in all cases where $\Sigma_c \subseteq \Sigma_m$ and Σ_a compares with both.

We do not know yet whether the technical answer we have provided to this problem can be extended to cope with more complex situations, such as for instance the case where $\Sigma_c \subseteq \Sigma_m$ and $\Sigma_a \subseteq \Sigma_m$ (the algorithm defined in 5.2.3 may not give the optimal supervisor in this case), or the case where one wants to enforce simultaneously the opacity of two predicates with respect to two attackers with different interfaces. An important question to be studied before applications are considered is the relation between opacity and finite state abstraction of possibly infinite state systems. Another topic of interest is the preservation of opacity by algebraic operations of system composition.

References

[1] Rajeev Alur, Pavol Černý, and Steve Zdancewic. Preserving secrecy under refinement. In *ICALP '06: Proceedings (Part II) of the 33rd International Colloquium on Automata, Languages and Programming*, pages 107–118. Springer, 2006.

[2] E. Badouel, M. Bednarczyk, A. Borzyszkowski, B. Caillaud, and P. Darondeau. Concurrent secrets. *Discrete Event Dynamic Systems*, 17:425–446, 2007. extended version of a paper presented at Wodes'06.

[3] R. D. Brandt, V. Garg, R. Kumar, F. Lin, S. I. Marcus, and W. M. Wonham. Formulas for calculating supremal controllable and normal sublanguages. *Systems & Control Letters*, 15(2):111–117, 1990.

[4] Jeremy Bryans, Maciej Koutny, Laurent Mazaré, and Peter Y. A. Ryan. Opacity generalised to transition systems. In Theo Dimitrakos, Fabio Martinelli, Peter Y. A. Ryan, and Steve A. Schneider, editors, *Revised Selected Papers of the 3rd International Workshop on Formal Aspects*

in Security and Trust (FAST'05), volume 3866 of *Lecture Notes in Computer Science*, pages 81–95, Newcastle upon Tyne, UK, 2006. Springer.

[5] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems.* Kluwer Academic Publishers, 1999.

[6] J. Dubreil, T. Jéron, and H. Marchand. Construction de moniteurs pour la surveillance de propriétés de sécurité. In *6ème Colloque Francophone sur la Modélisation des Systèmes Réactifs*, Lyon, France, October 2007.

[7] J. Ligatti, L. Bauer, and D. Walker. Edit automata: enforcement mechanisms for run-time security policies. *Int. J. Inf. Sec.*, 4(1-2):2–16, 2005.

[8] L. Mazaré. Using unification for opacity properties. In *Proceedings of the 4th IFIP WG1.7 Workshop on Issues in the Theory of Security (WITS'04)*, pages 165–176, Barcelona (Spain), 2004.

[9] Fred B. Schneider. Enforceable security policies. *ACM Trans. Inf. Syst. Secur.*, 3(1):30–50, 2000.

[10] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.